Splits the string into an array, reverses it, and joins it back.

2. Check for Palindrome

```
function isPalindrome(str) {
  const reversed = str.split('').reverse().join('');
  return (str === reversed) ? "Yes, it is" : "No, it isn't";
}
console.log(isPalindrome("121")); // Output: "Yes, it is"
console.log(isPalindrome("911")); // Output: "No, it isn't"
```

A **palindrome** reads the same backward as forward.

3. Generate Fibonacci Series

```
function fibonacci(n) {
  const sequence = [0, 1];
  for (let i = 2; i < n; i++) {
    sequence.push(sequence[i-1] + sequence[i-2]);
  }
  return sequence.slice(0, n);
}
console.log(fibonacci(10));
```

Generates the first `n` numbers of the Fibonacci sequence.

4. Find Factorial of a Number

```
function factorial(n) {
  if (n < 0) return "Factorial undefined for negatives.";
  let result = 1;
  for (let i = 1; i <= n; i++) {
    result *= i;
  }
  return result;
}
console.log(factorial(5)); // Output: 120
```

Factorial grows rapidly; use recursion cautiously.

5. Check for Prime Number

```
function isPrime(num) {
  if (num < 2) return false;
  for (let i = 2; i <= Math.sqrt(num); i++) {
    if (num % i === 0) return false;
  }
  return true;
}
console.log(isPrime(17)); // Output: true
```

Only divisible by 1 and itself.

6. Count Vowels and Consonants

```
function countLetters(str) {
  const vowels = str.match(/[aeiou]/gi) || [];
  const consonants = str.match(/[bcdfghjklmnpqrstvwxyz]/gi) || [];
  return `Vowels: ${vowels.length}, Consonants: ${consonants.length}`;
}
console.log(countLetters("Ganesh205"));
```

7. Sort an Array Numerically

```
function sortArray(arr) {
  return arr.sort((a, b) => a - b);
}
console.log(sortArray([10, 1, 4, 2, 3]));
```

8. Merge Two Arrays

```
function mergeArrays(arr1, arr2) {
  return [...arr1, ...arr2];
}
console.log(mergeArrays([1, 2, 3], [4, 5, 6]));
```

9. Find Largest Element

```
function findLargest(arr) {
  return Math.max(...arr);
}
console.log(findLargest([1, 2, 15, 4, 5]));
```

10. Remove Duplicates from Array

```
function removeDuplicates(arr) {
  return [...new Set(arr)];
}
console.log(removeDuplicates([1, 2, 2, 3, 4, 4, 5]));
```

11. Check Armstrong Number

```
function isArmstrong(n) {
  const digits = n.toString().split('');
  const sum = digits.reduce((acc, digit) => acc + Math.pow(+digit, digits.length),
0);
  return (sum === n) ? "Yes, it is" : "It is not";
}
console.log(isArmstrong(153));
```

12. Reverse a Number

```
function reverseNumber(n) {
  const reversed = parseInt(n.toString().split('').reverse().join('')) *
Math.sign(n);
  return reversed;
}
console.log(reverseNumber(119)); // Output: 911
```

13. Calculate GCD

```
function gcd(a, b) {
  if (!b) return a;
  return gcd(b, a % b);
}
console.log(gcd(12, 8)); // Output: 4
```

14. Anagram Checker

```
function isAnagram(str1, str2) {
  const normalize = str => str.replace(/\s/g,
'').toLowerCase().split('').sort().join('');
  return normalize(str1) === normalize(str2);
}
console.log(isAnagram("sIl e nt", "listen"));
```

15. Count Digits

```
function countDigits(num) {
  return num.toString().replace('-', '').length;
}
console.log(countDigits(911));
```

16. Prime Numbers in a Range

```
function primesInRange(start, end) {
  const primes = [];
  for (let i = start; i <= end; i++) {
    if (isPrime(i)) primes.push(i);
  }
  return primes;
}
console.log(primesInRange(10, 30));
```

17. Second Largest Element

```
function secondLargest(arr) {
  const unique = [...new Set(arr)];
  unique.sort((a, b) => b - a);
  return unique[1];
}
console.log(secondLargest([1, 3, 5, 8, 2, 9]));
```

18. Pascal's Triangle

```
function generatePascal(n) {
  const triangle = [];
  for (let row = 0; row < n; row++) {
    triangle[row] = [1];
    for (let col = 1; col < row; col++) {
      triangle[row][col] = triangle[row - 1][col - 1] + triangle[row - 1][col];
    }
    if (row) triangle[row].push(1);
  }
  return triangle.map(row => row.join(' ')).join('\n');
}
console.log(generatePascal(5));
```

19. Find Missing Number in Array

```
function findMissing(arr, n) {
  const expectedSum = (n * (n + 1)) / 2;
  const actualSum = arr.reduce((acc, num) => acc + num, 0);
  return expectedSum - actualSum;
}
console.log(findMissing([1, 2, 4, 5, 6], 6)); // Output: 3
```

Uses the formula for the sum of the first `n` natural numbers.

20. Check if Two Strings Are Isomorphic

```
function isIsomorphic(s, t) {
  if (s.length !== t.length) return false;

  const mapS = new Map();
  const mapT = new Map();

  for (let i = 0; i < s.length; i++) {
    if (!mapS.has(s[i])) mapS.set(s[i], t[i]);
    if (!mapT.has(t[i])) mapT.set(t[i], s[i]);
    if (mapS.get(s[i]) !== t[i] || mapT.get(t[i]) !== s[i]) return false;
  }
  return true;
}
console.log(isIsomorphic("egg", "add")); // Output: true
```

Maintains a one-to-one mapping between characters.

21. Move Zeros to End

```
function moveZeros(arr) {
  const nonZero = arr.filter(x => x !== 0);
  const zeros = arr.filter(x => x === 0);
  return [...nonZero, ...zeros];
}
console.log(moveZeros([0,1,0,3,12])); // Output: [1,3,12,0,0]
```

22. Find Intersection of Two Arrays

```
function arrayIntersection(arr1, arr2) {
  return arr1.filter(value => arr2.includes(value));
}
console.log(arrayIntersection([1,2,2,1], [2,2])); // Output: [2,2]
```

23. Find First Non-Repeating Character

```
function firstNonRepeatingChar(str) {
  for (let char of str) {
    if (str.indexOf(char) === str.lastIndexOf(char)) return char;
  }
  return null;
}
console.log(firstNonRepeatingChar("aabbcdd")); // Output: 'c'
```

24. Sum of Elements in Array

```
function arraySum(arr) {
  return arr.reduce((sum, num) => sum + num, 0);
}
console.log(arraySum([1,2,3,4,5])); // Output: 15
```

25. Check if Number is Power of Two

```
function isPowerOfTwo(n) {
  return n > 0 && (n & (n - 1)) === 0;
}
console.log(isPowerOfTwo(16)); // Output: true
```

Bit manipulation trick: only powers of two have exactly one bit set.

26. Find Duplicate Number

```
function findDuplicate(nums) {
  const set = new Set();
  for (let num of nums) {
    if (set.has(num)) return num;
    set.add(num);
  }
}
console.log(findDuplicate([1,3,4,2,2])); // Output: 2
```

27. Longest Word in a String

```
function findLongestWord(str) {
  return str.split(' ').sort((a, b) => b.length - a.length)[0];
}
console.log(findLongestWord("The quick brown fox jumped over the lazy dog"));
```

28. Capitalize First Letter of Each Word

```
function capitalizeWords(str) {
  return str.split(' ').map(word => word.charAt(0).toUpperCase() +
word.slice(1)).join(' ');
}
console.log(capitalizeWords("hello world"));
```

29. Flatten a Nested Array

```
function flattenArray(arr) {
  return arr.flat(Infinity);
}
console.log(flattenArray([1, [2, [3, [4]]]])); // Output: [1,2,3,4]
```

30. Find Majority Element

```
function majorityElement(nums) {
  const count = {};
  const n = nums.length;
  for (let num of nums) {
    count[num] = (count[num] || 0) + 1;
    if (count[num] > n/2) return num;
  }
}
console.log(majorityElement([2,2,1,1,2,2])); // Output: 2
```

31. Find Subsets of an Array

```
function subsets(arr) {
  const result = [[]];
  for (let num of arr) {
    const len = result.length;
    for (let i = 0; i < len; i++) {
      result.push([...result[i], num]);
    }
  }
  return result;
}
console.log(subsets([1,2]));
```

32. Debounce Function

```
function debounce(func, delay) {
  let timer;
  return function(...args) {
    clearTimeout(timer);
    timer = setTimeout(() => func.apply(this, args), delay);
  };
}
const processChange = debounce(() => console.log('Saved!'), 300);
```

Debouncing delays execution until a pause happens.

33. Throttle Function

```
function throttle(func, limit) {
  let inThrottle;
  return function(...args) {
    if (!inThrottle) {
      func.apply(this, args);
      inThrottle = true;
      setTimeout(() => inThrottle = false, limit);
    }
  };
}
const handleScroll = throttle(() => console.log('Scrolling!'), 200);
```

34. Deep Clone an Object

```
function deepClone(obj) {
  return JSON.parse(JSON.stringify(obj));
}
const original = { a: 1, b: { c: 2 } };
const clone = deepClone(original);
console.log(clone);
```

35. Group Anagrams

```
function groupAnagrams(words) {
  const map = new Map();
  for (let word of words) {
    const sorted = word.split('').sort().join('');
    map.set(sorted, [...(map.get(sorted) || []), word]);
  }
  return Array.from(map.values());
}
console.log(groupAnagrams(["eat","tea","tan","ate","nat","bat"]));
```

36. Validate Brackets

```
function isValidBrackets(s) {
  const stack = [];
  const map = {')':'(', ']':'[', '}':'{'};
  for (let char of s) {
    if (['(','{','['].includes(char)) {
      stack.push(char);
    } else {
      if (stack.pop() !== map[char]) return false;
    }
  }
  return stack.length === 0;
}
console.log(isValidBrackets("()[]{}")); // Output: true
```

37. Find All Permutations

```
function permutations(str) {
  if (str.length <= 1) return [str];
  let result = [];
  for (let i = 0; i < str.length; i++) {
    const char = str[i];
    const perms = permutations(str.slice(0, i) + str.slice(i + 1));
    for (let perm of perms) {
      result.push(char + perm);
    }
  }
  return result;
}
console.log(permutations("abc"));
```

38. Implement Currying

```
function curry(fn) {
  return function curried(...args) {
    if (args.length >= fn.length) {
      return fn(...args);
    } else {
      return (...nextArgs) => curried(...args, ...nextArgs);
    }
  };
}
function add(a, b, c) {
  return a + b + c;
}
const curriedAdd = curry(add);
console.log(curriedAdd(1)(2)(3)); // Output: 6
```